PATENT
Attorney Docket No. 0023-0209

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re Patent Application of	)	
	)	
Nir Zuk et al.	)	Group Art Unit: 2137
	)	
Application No.: 10/072,683	)	Examiner: M. Nguyen
	)	
Filed: February 8, 2002	)	
	)	
For: MULTI-METHOD GATEWAY-BASED NETWORK SECURITY SYSTEMS AND METHODS	)	

APPEAL BRIEF

U.S. Patent and Trademark Office
Customer Window, Mail Stop Appeal Brief – Patents
Randolph Building
401 Dulany Street
Alexandria, Virginia 22314

Sir:

This Appeal Brief is submitted in response to the Final rejection mailed March 28, 2007 and in support of the Notice of Appeal filed June 28, 2007.

I. REAL PARTY IN INTEREST

The real party in interest in this appeal is Juniper Networks, Inc.

II. RELATED APPEALS AND INTERFERENCES

The Appellants are unaware of any related appeals, interferences or judicial proceedings.

III. STATUS OF CLAIMS

Claims 1-7, 10, 12-18, 21-25, 27, 31-33, 35, 37-46, 49, 50 and 52-69 are pending in this application. Claims 8, 9, 19, 20, 26, 28-30, 34, 36, 47, 48 and 51 have been previously canceled without prejudice or disclaimer. Claims 1-7, 10, 12-18, 21, 23-25, 27, 31-33, 35, 37, 38, 40 and 41, have been rejected under 35 U.S.C. § 103(a) as being unpatentable over Gleichauf et al. (U.S. Patent No. 6,499,107; hereinafter Gleichauf '107) and Gleichauf et al. (U.S. Patent No. 6,324,656; hereinafter Gleichauf '656) in view of Nikander et al. (U.S. Patent No. 6,253,321; hereinafter Nikander), in view of Copeland III (U.S. Patent Application Publication No. 2003/0105976; hereinafter Copeland III) and further in view of Alexander et al. (U.S. Patent Publication No. 2004/0258073; hereinafter Alexander); claims 22 and 39 have been rejected under 35 U.S.C. § 103(a) as being unpatentable over Gleichauf '107 and Gleichauf '656 in view of Nikander; and claims 42-46, 49, 50 and 52-69 have been rejected under 35 U.S.C. § 103(a) as being unpatentable over Gleichauf '107 in view of Nikander in view of Trcka et al. (U.S. Patent No. 6,453,345; hereinafter Trcka) in view of Copeland III and further in view of Alexander. Claims 1-7, 10, 12-18, 21-25, 27, 31-33, 35, 37-46, 49, 50 and 52-69 are the subject of the present appeal.

IV. STATUS OF AMENDMENTS

No Amendment has been filed subsequent to the Final Office Action mailed March 28, 2007.

V. SUMMARY OF THE CLAIMED SUBJECT MATTER

Each of the independent claims involved in this appeal is recited below, followed in parenthesis by examples of where support can be found in the specification and drawings for the claimed subject matter. In addition, each dependent claim argued separately below is also summarized in a similar manner.

Claim 1 recites: A method for detecting and preventing security breaches in a network, the method comprising: reassembling a plurality of TCP packets in network traffic into a TCP stream (e.g., page 41, lines 18-21; Fig. 13, step 350); grouping the plurality of TCP packets into packet flows and sessions (e.g., page 41, lines 21-24; Fig. 13, step 355); storing the packet flows in packet flow descriptors, the packet flow descriptors being addressed by a hash value computed from a 5-tuple comprising a source IP address, a destination IP address, a source port, a destination port and a protocol type (e.g., page 33, lines 10-29; Fig. 7, flow table 155); inspecting the TCP stream to detect information indicative of a security breach (e.g., page 41, lines 27-29; Fig. 13, step 370); dropping a TCP packet from the TCP stream if the TCP stream contains information indicative of a security breach (e.g., page 41, lines 29-30; Fig. 13, step 380); forwarding a TCP packet from the TCP stream to a network destination if the TCP stream does not contain information indicative of a security breach (e.g., page 42, lines 21-25; Fig. 13, step 410), wherein inspecting the TCP stream to detect information indicative of a security breach comprises: storing a plurality of protocol specifications supported by the network in a protocol database (e.g., page 37, lines 14-21; Fig. 10, table 245), querying the protocol database to determine whether the plurality of TCP packets are compliant with one or more of the plurality of

protocol specifications in the protocol database (e.g., page 36, line 30 to page 37, line 7; Fig. 9, step 225), and searching for a network attack identifier in the TCP stream based on the packet flow descriptors and sessions associated with the TCP stream (e.g., page 42, lines 10-14; Fig. 13, step 385).

Claim 18 recites: A method comprising: reassembling a plurality of TCP packets into a TCP stream (e.g., page 41, lines 18-21; Fig. 13, step 350); inspecting the TCP stream to detect information indicative of a security breach (e.g., page 41, lines 27-29; Fig. 13, step 370); dropping a TCP packet from the TCP stream if the TCP stream contains information indicative of a security breach (e.g., page 41, lines 29-30; Fig. 13, step 380); forwarding a TCP packet from the TCP stream to a network destination if the TCP stream does not contain information indicative of a security breach (e.g., page 42, lines 21-25; Fig. 13, step 410); and grouping the plurality of TCP packets into packet flows and sessions, wherein grouping the plurality of TCP packets into packet flows and sessions comprises storing the packet flows and sessions in a hash table (e.g., page 33, lines 10-29; Fig. 7, flow table 155), wherein inspecting the TCP stream to detect information indicative of a security breach comprises: storing the packet flows in packet flow descriptors (e.g., page 33, lines 10-29) , and searching for a network attack identifier in the TCP stream based on the packet flow descriptors and sessions associated with the TCP stream (e.g., page 42, lines 10-14; Fig. 13, step 385), wherein storing the packet flows and sessions in a hash table comprises computing a hash value from a 5-tuple comprising a source IP address, a destination IP address, a source port, a destination port and a protocol type (e.g., page 33, lines 10-29).

Claim 22 recites: A method for detecting and preventing security breaches in a network, the method comprising: reassembling a plurality of TCP packets into a TCP stream (e.g., page 41, lines 18-21; Fig. 13, step 350); inspecting the TCP stream to detect information indicative of a security breach (e.g., page 41, lines 27-29; Fig. 13, step 370); dropping a TCP packet from the TCP stream if the TCP stream contains information indicative of a security breach (e.g., page 41, lines 29-30; Fig. 13, step 380); forwarding a TCP packet from the TCP stream to a network destination if the TCP stream does not contain information indicative of a security breach (e.g., page 42, lines 21-25; Fig. 13, step 410), wherein inspecting the TCP stream to detect information indicative of a security breach comprises: querying a signatures database to determine whether there are matching signatures in the TCP stream using deterministic finite automata for pattern matching (e.g., page 39, lines 3-24; Fig. 11, step 275).

Claim 24 recites: A system for detecting and preventing security breaches in a network, the system comprising: a TCP reassembly software module for reassembling a plurality of TCP packets in network traffic into a TCP stream (e.g., page 32, lines 10-12; Fig. 6, 115); a software module for inspecting the TCP stream to detect information indicative of a security breach (e.g., page 36, lines 27-30; Fig. 6, 130); a software module for dropping a TCP packet from the TCP stream if the TCP stream contains information indicative of a security breach (e.g., page 41, lines 29-30; Fig. 6, 130); a software module for forwarding a TCP packet from the TCP stream to a network destination if the TCP stream does not contain information indicative of a security breach (e.g., page 42, lines 21-25; Fig. 6, 150); and at least one processing device configured to execute the TCP reassembly software module, the software module for inspecting the TCP

stream, the software module for dropping a TCP packet and the software module for forwarding a TCP packet (page 27, lines 4-26; Fig. 2, server 30), wherein the software module for inspecting the TCP stream comprises at least a protocol anomaly detection software module (Fig. 6, 130) and a flow manager software module (Fig. 6, 120), the protocol anomaly detection software module comprising: a routine for storing a plurality of protocol specifications supported by the network in a protocol database (e.g., page 37, lines 14-21), and a routine for querying the protocol database to determine whether the plurality of TCP packets are compliant with one or more of the plurality of protocol specifications in the protocol database (e.g., page 36, line 30 to page 37, line 7), wherein the flow manager software module is configured to: group the plurality of TCP packets into packet flows and sessions (e.g., page 41, lines 21-24), store the packet flows in packet flow descriptors, the packet flow descriptors being addressed by a hash value computed from a 5-tuple comprising a source IP address, a destination IP address, a source port, a destination port and a protocol type, and search for a network attack identifier in the TCP stream based on the packet flow descriptors and sessions associated with the TCP stream (e.g., page 33, lines 10-29).

Claim 27 recites: A system comprising: a TCP reassembly software module for reassembling a plurality of TCP packets network traffic into a TCP stream (e.g., page 32, lines 10-12; Fig. 6, 115); a software module for inspecting the TCP stream to detect information indicative of a security breach (e.g., page 36, lines 27-30; Fig. 6, 130); a software module for dropping a TCP packet from the TCP stream if the TCP stream contains information indicative of a security breach (e.g., page 41, lines 29-30; Fig. 6, 130); a software module for forwarding a

TCP packet from the TCP stream to a network destination if the TCP stream does not contain information indicative of a security breach (e.g., page 42, lines 21-25; Fig. 6, 150); a flow manager software module for grouping the plurality of TCP packets into packet flows and sessions, wherein the flow manager software module comprises a routine for storing the packet flows and sessions into a hash table (e.g., page 33, lines 10-14), storing the packet flows in packet flow descriptors, and searching for a network attack identifier in the TCP stream based on the packet flow descriptors and sessions associated with the TCP stream (e.g., page 41, line 31 to page 42, line 3) and at least one processing device configured to execute the TCP reassembly software module (e.g., page 27, lines 4-15; Fig. 2, server 30), the software module for inspecting the TCP stream, the software module for dropping a TCP packet, a software module for forwarding a TCP packet and the flow manager software module, wherein the routine for storing the packet flows and sessions into a hash table comprises a routine for computing a hash value from a 5-tuple comprising a source IP address, a destination IP address, a source port, a destination port and a protocol type (e.g., page 33, lines 10-29).

Claim 39 recites: A system for detecting and preventing security breaches in a network, the system comprising: a TCP reassembly software module for reassembling a plurality of TCP packets in network traffic into a TCP stream (e.g., page 41, lines 18-21; Fig. 6, 115); a software module for inspecting the TCP stream to detect information indicative of a security breach (e.g., page 41, lines 27-29; Fig. 6, 130); a software module for dropping a TCP packet from the TCP stream if the TCP stream contains information indicative of a security breach (e.g., page 41, lines 29-30; Fig. 6, 130); a software module for forwarding a TCP packet from the TCP stream to a

network destination if the TCP stream does not contain information indicative of a security breach (e.g., page 42, lines 21-25; Fig. 6, 150), wherein the software module for inspecting the TCP stream to detect information indicative of a security breach comprises a stateful signature detection software module (Fig. 6, 135), the stateful signature detection software module comprising: a routine for querying a signatures database to determine whether there are matching attack signatures in the TCP stream using deterministic finite automata for pattern matching (e.g., page 39, lines 3-24); and at least one processing device configured to execute the TCP reassembly software module, the software module for inspecting the TCP stream, the software module for dropping a TCP packet and the software module for forwarding a TCP packet (e.g., page 27, lines 4-15; Fig. 2, server 30).

Claim 42 recites: A system for detecting and preventing security breaches in a network, the system comprising: a network intrusion detection and prevention sensor located in a network gateway (e.g., page 26, lines 4-15; Fig. 2, 25a-d), wherein the network intrusion detection and prevention sensor including at least one processor configured to execute (e.g., page 27, lines 4-16; server 30): a routine for reassembling a plurality of TCP packets into a TCP stream (e.g., page 41, lines 18-21); a software module for inspecting the TCP stream to detect information indicative of a security breach (e.g., page 41, lines 27-29), wherein inspecting the TCP stream to detect information indicative of a security breach comprises: storing a plurality of protocol specifications supported by the network in a protocol database, and querying the protocol database to determine whether the plurality of TCP packets are compliant with one or more of the plurality of protocol specifications in the protocol database (e.g., page 36, line 27 to page 37,

line 7); a software module for dropping a TCP packet from the TCP stream if the TCP stream contains information indicative of a security breach (e.g., page 41, lines 29-30); and a software module for forwarding a TCP packet from the TCP stream to a network destination if the TCP stream does not contain information indicative of a security breach (e.g., page 42, lines 21-25); a central management server to control the network intrusion detection and prevention sensor (e.g., page 26, lines 12-17; Fig. 2, 30); and a graphical user interface (e.g., page 26, lines 17-22) for configuring the network intrusion detection and prevention sensor, wherein the network intrusion detection and prevention sensor further comprises a flow manager software module and a traffic signature detection module, the flow manager software module being configured to: group the plurality of TCP packets into packet flows and sessions (e.g., page 41, lines 21-24), and store the packet flows in packet flow descriptors, the packet flow descriptors being addressed by a hash value computed from a 5-tuple comprising a source IP address, a destination IP address, a source port, a destination port and a protocol type (e.g., page 33, lines 10-29), and the traffic signature detection module is configured to: search for a network attack identifier in the TCP stream based on the packet flow descriptors and sessions associated with the TCP stream (e.g., page 41, line 31 to page 42, line 3; Fig. 6, 140).

Claim 57 recites: A network intrusion detection and prevention sensor for detecting and preventing network security breaches at a network gateway (e.g., page 26, lines 4-15; Fig. 2, 25), the network intrusion detection and prevention sensor comprising: a flow manager software module for grouping a plurality of packets into packet flows and sessions (e.g., page 41, lines 21-24; Fig. 6, 130); a TCP reassembly software module for reassembling a plurality of TCP packets

from the plurality of packets into a TCP stream (e.g., page 41, lines 18-21; Fig. 6, 115); a software module for inspecting the TCP stream according to the packet flows and sessions to detect information indicative of a security breach (e.g. page 41, lines 27-29; Fig. 6, 130), wherein inspecting the TCP stream to detect information indicative of a security breach comprises: storing a plurality of protocol specifications supported by the network in a protocol database (e.g., page 37, lines 14-21), and querying the protocol database to determine whether the plurality of TCP packets are compliant with one or more of the plurality of protocol specifications in the protocol database (e.g., page 36, line 30 to page 37, line 7); a software module for dropping a packet from the plurality of packets if the TCP stream contains information indicative of a security breach (e.g., page 41, lines 29-30; Fig. 6, 130); a software module for forwarding a packet from the plurality of packets to a network destination if the TCP stream does not contain information indicative of a security breach (e.g., page 42, lines 21-25; Fig. 6, 150); a software module for grouping the plurality of TCP packets into packet flows and sessions and storing the packet flows in packet flow descriptors, the packet flow descriptors being addressed by a hash value computed from a 5-tuple comprising a source IP address, a destination IP address, a source port, a destination port and a protocol type; a software module for searching for a network attack identifier in the TCP stream based on the packet flow descriptors and sessions associated with the TCP stream (e.g., page 33, lines 10-29); and at least one processing device configured to execute the flow manager software module, the TCP reassembly software module, the software module for inspecting the TCP stream, the software module for dropping a packet, a software module for forwarding a packet, a software module for grouping the plurality of TCP packets and the software module for searching for a network attack identifier (e.g., page 27, lines 3-15; Fig. 2,

30).

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

A. Claims 1-7, 10, 12-18, 21, 23-25, 27, 31-33, 35, 37, 38, 40 and 41, have been rejected under 35 U.S.C. § 103(a) as being unpatentable over Gleichauf '107 and Gleichauf '656 in view of Nikander in view of Copeland III and further in view of Alexander.

B. Claims 22 and 39 have been rejected under 35 U.S.C. § 103(a) as being unpatentable over Gleichauf '107 and Gleichauf '656 in view of Nikander.

C. Claims 42-46, 49, 50 and 52-69 have been rejected under 35 U.S.C. § 103(a) as being unpatentable over Gleichauf '107 in view of Nikander in view of Trcka in view of Copeland III and further in view of Alexander.

VII. ARGUMENT

A. Rejection under 35 U.S.C. § 103 based on Gleichauf '107 and Gleichauf '656 in view of Nikander in view of Copeland III and further in view of Alexander

The initial burden of establishing a *prima facie* basis to deny patentability to a claimed invention always rests upon the Examiner. In re Oetiker, 977 F.2d 1443, 24 USPQ2d 1443 (Fed. Cir. 1992). In rejecting a claim under 35 U.S.C. § 103, the Examiner must provide a factual basis to support the conclusion of obviousness. In re Warner, 379 F.2d 1011, 154 USPQ 173 (CCPA 1967). Based upon the objective evidence of record, the Examiner is required to make the factual inquiries mandated by Graham v. John Deere Co., 86 S.Ct. 684, 383 U.S. 1, 148 USPQ 459 (1966). The Examiner is also required to explain how and why one having ordinary

skill in the art would have been realistically motivated to modify an applied reference and/or combine applied references to arrive at the claimed invention. Uniroyal, Inc. v. Rudkin-Wiley Corp., 837 F.2d 1044, 5 USPQ2d 1434 (Fed. Cir. 1988).

1. Claims 1-7, 10, 12-18, 21, 23-25, 27, 31-33, 35, 37, 38, 40 and 41

With these principles in mind, claim 1 recites a method for detecting and preventing security breaches in a network. The method includes grouping the plurality of TCP packets into packet flows and sessions and storing the packet flows in packet flow descriptors. Claim 1 also recites inspecting the TCP stream to detect information indicative of a security breach, wherein the inspecting comprises searching for a network attack identifier in the TCP stream based on the packet flow descriptors and sessions associated with the TCP stream. Claim 1 further recites that the packet flow descriptors are addressed by a hash value computed from a 5-tuple comprising a source IP address, a destination IP address, a source port, a destination port and a protocol type.

The Final Office Action admits that none of Gleichauf '107, Gleichauf '656, Nikander or Copeland III discloses this latter feature (Final Office Action – page 6). The Final Office Action, however, states that Alexander discloses computing a hash value from a 5-tuple comprising a source IP address, a destination IP address, a source port, a destination port and a protocol type and points to Alexander at page 3, paragraph 27 for support (Final Office Action – page 6).

Alexander discloses that mapping function services 230 performs a hash function on the 5-tuple for the purpose of performing multi-protocol label switching (MPLS) (Alexander – Abstract and paragraphs 24-27). Alexander is not at all related to detecting security breaches by storing packet flows in packet flow descriptors that are addressed by a hash value computed from

a 5-tuple comprising a source IP address, a destination IP address, a source port, a destination port and a protocol type, as required by claim 1. Alexander, therefore, cannot further disclose searching for a network attack identifier in a TCP stream <u>based on the packet flow descriptors</u> and sessions associated with the TCP stream, where the packet flow descriptors are addressed by the hash value computed from the 5-tuple discussed above, as required by claim 1. In contrast, Alexander merely discloses using a mapping function for performing MPLS switching.

For at least these reasons, the combination Gleichauf '107, Gleichauf '656, Nikander, Copeland III and Alexander does not disclose or suggest each of the features of claim 1.

In addition, even if, for the sake of argument, the combination of these five references could be construed to disclose or suggest each of the features of claim 1, the Appellants respectfully assert that the motivation to combine these five references does not satisfy the requirements of 35 U.S.C. § 103.

For example, as discussed above, Alexander is not at all related to detecting security breaches and is not at all related to the other four references. That is, Gleichauf '107 is directed to adaptive network security (Gleichauf '107 – Abstract), Gleichauf '656 is directed to performing network vulnerability assessment (Gleichauf '656 – Abstract), Nikander is directed to a data processing system implementing an IP security protocol (Nikander – Abstract), and Copeland III is directed to a flow-based intrusion detection system (Copeland III – Abstract). These four references are directed to solving different problems in significantly different ways and are essentially unrelated, other than the fact that each of the references tangentially relates to some aspect of network security. Alexander, in contrast to these four references, is directed to load sharing in an MPLS system, which is totally unrelated to any of the other four references.

The Appellants respectfully assert that it would not have been obvious to combine Alexander with the other four references due to the disparate nature of these references.

For example, the Final Office Action states that it would have been obvious to combine Alexander with Gleichauf '107, Gleichauf '656, Nikander and Copeland III "so as to effectively performing packet filtering" (Final Office Action – page 6). This alleged motivation is merely a conclusory statement providing an alleged benefit of the combination. Such motivation does not satisfy the requirements of 35 U.S.C. § 103. In addition, the Appellants note that no portion of any of the references is pointed to as providing objective motivation for combining Alexander, which is directed to load sharing in an MPLS system, with the combination of the other four references. The Appellants respectfully assert that it would not have been obvious to combine these unrelated references absent impermissible hindsight.

In response to similar arguments made in the response filed January 17, 2007, the Final Office Action states that Alexander is directed to data networking and distributing data flows and therefore, it is proper to combine Alexander with other references also dealing with data communication networking (Final Office Action – page 2). The Appellants respectfully disagree.

The mere fact that one reference (such as Alexander) or a number of references (such as Gleichauf '656, Nikander, Copeland III and Alexander) allegedly provide some missing disclosure with respect to a claim does not satisfy the requirements of 35 U.S.C. § 103 as to why it would have been obvious to combine the references. In this case, the Appellants respectfully assert that it would not have been obvious to combine these Alexander with the other four references without the benefit of the Appellants' disclosure.

In addition, the alleged motivation for combining Gleichauf '656, Nikander and Copeland

III with Gleichauf '107 also do not satisfy the requirements of 35 U.S.C. § 103. For example, the Final Office Action has not provided any statement regarding why it would have been obvious to combine Gleichauf '107 and Gleichauf '656. The mere fact that the two references share a common inventor does not mean that the references are properly combinable.

Further, the alleged motivation for combining Nikander with the combination of Gleichauf '107 and Gleichauf '656 (i.e., to effectively manage communications data – Final Office Action – page 5) is merely a conclusory statement providing an alleged benefit of the combination. Such motivation does not satisfy the requirements of 35 U.S.C. § 103.

In addition, the alleged motivation for combining Copeland III with the combination of Gleichauf '107, Gleichauf '656 and Nikander (i.e., to effectively determine if the traffic data appears to be legitimate or possible suspicious activity – Final Office Action – pages 5-6) is once again merely a conclusory statement providing an alleged benefit of the combination. Such motivation does not satisfy the requirements of 35 U.S.C. § 103.

Appellants respectfully assert that the mere fact that each of these four references may be tangentially related in one minor aspect of their otherwise diverse disclosures, such as involving some aspect of network security, does not mean that the references themselves are directed to analogous art or that one of ordinary skill in the art would look to combine portions of these references. Once again, the mere fact that one or more references allegedly provide some missing disclosure with respect to a claim does not satisfy the requirements of 35 U.S.C. § 103 as to why it would have been obvious to combine the references. The Appellant asserts that one of ordinary skill in the art would not have looked to combine features from these four references

(and then combine these four references with Alexander) due to the disparate nature of the subject matter of these references.

For at least the reasons discussed above, the Appellants respectfully submit that the imposed rejection of claim 1 under 35 U.S.C. § 103 based on the combination of Gleichauf '107, Gleichauf '656, Nikander, Copeland III and Alexander is improper. Accordingly, reversal of the rejection of claims 1-7, 10, 12-18, 21, 23-25, 27, 31-33, 35, 37, 38, 40 and 41 is respectfully requested.

B. Rejection under 35 U.S.C. § 103 based on Gleichauf '107 and Gleichauf '656 in view of Nikander.

1. Claims 22 and 39

Claim 22 recites a method for detecting and preventing security breaches in a network. The method includes querying a signatures database to determine whether there are matching signatures in the TCP stream using deterministic finite automata for pattern matching. The Final Office Action admits that Gleichauf '107, Gleichauf '656 and Nikander do not disclose these features, but takes Official Notice that employing "use of deterministic finite automaton for providing a pattern matching is well known in the theory of computation" (Final Office Action – page 13). The Appellants, however, note that claim 22 recites more than just a generic use of pattern matching using deterministic finite automata. For example, claim 22 recites querying a signatures database to determine whether there are matching signatures in the TCP stream using deterministic finite automata for pattern matching. Therefore, the mere fact that deterministic finite automata techniques are known, in general, does not mean that they are well know in the

manner recited in claim 22.

In response to a similar argument regarding the use of Official Notice, the Office Action mailed October 17, 2006 cited two documents (listed on the PTO-892 accompanying the Office Action mailed October 17, 2006 and referred to herein as Navarro 1997 and Navarro 1998) that allegedly disclose the use of deterministic finite automata (DFA) (Office Action mailed October 17, 2006 – page 3). As pointed out in the response filed January 17, 2007, the Appellants noted that Navarro 1997 and Navarro 1998 may generally disclose DFA techniques. These documents, however, do not disclose the use of DFA in the manner recited in claim 22. The Appellants, therefore, requested that any subsequent communication point to a reference that discloses the specifically claimed features or withdraw the rejection.

In response to these arguments made in the previous response, the Final Office Action states that claim 22 is addressed by the combination of Gleichauf and Nikander and maintains that DFA is well known in solving pattern matching problems (Final Office Action – pages 2-3). As discussed above, claim 22 recites more than just a generic use of pattern matching using deterministic finite automata. That is, claim 22 specifically recites <u>querying a signatures database to determine whether there are matching signatures in the TCP stream using deterministic finite automata for pattern matching</u>. The mere fact that DFA is known, in general, does not mean that they are well known in the manner recited in claim 22. The Appellants also note that the Final Office Action has not provided any additional support regarding the subject of the Official Notice (e.g., why such a feature would have been obvious to use in the manner recited in claim 22 or why it would have been obvious to use such a feature in the combination of Gleichauf '107, Gleichauf '656 and Nikander). Accordingly, the Appellants respectfully assert

that a prima facie case of obviousness under 35 U.S.C. § 103 has not been established.

For at least these reasons, the combination of Gleichauf '107, Gleichauf '656 and Nikander does not disclose or suggest each of the features of claim 22.

In addition, even if, for the sake of argument, the combination of Gleichauf '107, Gleichauf '656 and Nikander could be fairly construed to disclose or suggest each of the features of claim 22, the Appellants respectfully assert that the motivation for combining these references does not satisfy the requirements of 35 U.S.C. § 103.

For example, the Final Office Action has not provided any statement regarding why it would have been obvious to combine Gleichauf '107 and Gleichauf '656. The mere fact that the two references share a common inventor does not mean that the references are properly combinable.

In addition, the alleged motivation for combining Nikander with the combination of Gleichauf '107 and Gleichauf '656 (i.e., to effectively manage communications data – Final Office Action – page 13) is merely a conclusory statement providing an alleged benefit of the combination. Such motivation does not satisfy the requirements of 35 U.S.C. § 103.

Once again, the mere fact that one or more references allegedly provide some missing disclosure with respect to a claim does not satisfy the requirements of 35 U.S.C. § 103 as to why it would have been obvious to combine the references. The Appellants respectfully asserts that one of ordinary skill in the art would not have looked to combine features from these three references without the benefit of the Appellants' disclosure.

For at least the reasons discussed above, the Appellants respectfully submit that the imposed rejection of claim 22 under 35 U.S.C. § 103 based on the combination of Gleichauf

'107, Gleichauf '656 and Nikander is improper. Accordingly, reversal of the rejection of claims 22 and 39 is respectfully requested.

C. Rejection under 35 U.S.C. § 103 based on Gleichauf '107 in view of Nikander in view of Trcka in view of Copeland III and further in view of Alexander.

1. Claims 42-46, 49, 50 and 52-69

Claim 42 recites a system for detecting and preventing security breaches in a network. The system includes a network intrusion detection and prevention sensor that comprises a flow manager software module and a traffic signature detection module, the flow manager software module being configured to: group the plurality of TCP packets into packet flows and sessions, and store the packet flows in packet flow descriptors, the packet flow descriptors being addressed by a hash value computed from a 5-tuple comprising a source IP address, a destination IP address, a source port, a destination port and a protocol type, and the traffic signature detection module is configured to: search for a network attack identifier in the TCP stream based on the packet flow descriptors and sessions associated with the TCP stream.

Similar to the discussion above with respect to claim 1, the Final Office Action relies upon Alexander for allegedly disclosing packet flow descriptors being addressed by a hash value from a 5-tuple comprising a source IP address, a destination IP address, a source port, a destination port and a protocol type and points to Alexander at page 3, paragraph 27 for support (Final Office Action – page 17).

As discussed above with respect to claim 1, Alexander discloses that mapping function services 230 performs a hash function on the 5-tuple for the purpose of performing multi-

protocol label switching (MPLS) (Alexander – Abstract and paragraphs 24-27). Alexander is not at all related to network intrusion detection and prevention. More particularly, Alexander does not disclose or suggest <u>searching for a network attack identifier in a TCP stream based on the packet flow descriptors and sessions associated with the TCP stream</u>, where the packet flow descriptors are addressed by a hash value computed from a 5-tuple comprising a source IP address, a destination IP address, a source port, a destination port and a protocol type, as required by claim 42. In contrast, Alexander merely discloses using a mapping function for performing MPLS switching.

For at least these reasons, the combination Gleichauf '107, Nikander, Trcka, Copeland III and Alexander does not disclose or suggest each of the features of claim 42.

In addition, even if, for the sake of argument, the combination of these five references could be construed to disclose or suggest each of the features of claim 42, the Appellants respectfully assert that the motivation to combine these five references does not satisfy the requirements of 35 U.S.C. § 103.

For example, as discussed above, Alexander is not at all related to detecting security breaches and is not at all related to Gleichauf '107, Nikander and Copeland III. Trcka has been used in the rejection to allegedly disclose a network security system comprising a central management center and a graphical user interface for configuring the network intrusion detection and prevention sensor (Final Office Action – page 15). Trcka, however is also unrelated to the subject matter of Alexander. In addition, these four references (i.e., Gleichauf '107, Nikander, Trcka and Copeland III) are directed to solving different problems in significantly different ways and are essentially unrelated, other than the fact that each of the references tangentially relates to

some aspect of network security. Alexander, in contrast, is directed to load sharing in an MPLS system, which is totally unrelated to any of the other four references. The Appellants respectfully assert that it would not have been obvious to combine Alexander with the other four references due to the disparate nature of these references.

For example, the Final Office Action states that it would have been obvious to combine Alexander with Gleichauf '107, Nikander, Trcka and Copeland III "so as to effectively performing packet filtering" (Final Office Action – page 17). Similar to the discussion above with respect to claim 1, this alleged motivation is merely a conclusory statement providing an alleged benefit of the combination. Such motivation does not satisfy the requirements of 35 U.S.C. § 103.

Again, the mere fact that one reference (such as Alexander) or a number of references (such as Nikander, Trcka, Copeland III and Alexander) allegedly provide some missing disclosure with respect to a claim does not satisfy the requirements of 35 U.S.C. § 103 as to why it would have been obvious to combine the references. In this case, the Appellants respectfully assert that it would not have been obvious to combine these Alexander with the other four references without the benefit of the Appellants' disclosure.

In addition, the alleged motivation for combining Gleichauf '107, Nikander, Trcka and Copeland III also do not satisfy the requirements of 35 U.S.C. § 103. For example, the alleged motivation for combining Nikander with the combination of Gleichauf '107 (i.e., to effectively manage communications data – Final Office Action – page 15) is merely a conclusory statement providing an alleged benefit of the combination. The alleged motivation for combining Trcka with the combination of Gleichauf '107 and Nikander (i.e., to detect and protect against security

breaches, network failures and other types of data compromising events – Final Office Action – pages 15-16) is also merely a conclusory statement providing an alleged benefit of the combination. Further, the alleged motivation for combining Copeland III with the combination of Gleichauf '107, Nikander and Trcka (i.e., to effectively determine if the traffic data appears to be legitimate or possible suspicious activity – Final Office Action – page 16) is once again merely a conclusory statement providing an alleged benefit of the combination.

In summary, the alleged motivation for combining these disparate references amounts to conclusory statements providing alleged benefits of the combinations. The Appellants respectfully assert that such motivation does not satisfy the requirements of 35 U.S.C. § 103.

Once again, the Appellants respectfully assert that the mere fact that each of these four references may be tangentially related in one minor aspect of their otherwise diverse disclosures, such as involving some aspect of network security, does not mean that the references themselves are directed to analogous art or that one of ordinary skill in the art would look to combine portions of these references. The Appellants assert that one of ordinary skill in the art would not have looked to combine features from these four references (and then combine these four references with Alexander) due to the disparate nature of the subject matter of these references.

For at least the reasons discussed above, the Appellants respectfully submit that the imposed rejection of claim 42 under 35 U.S.C. § 103 based on the combination of Gleichauf '107, Nikander, Trcka, Copeland III and Alexander is improper. Accordingly, reversal of the rejection of claims 42-46, 49, 50 and 52-69 is respectfully requested.

VIII. CONCLUSION

In view of the foregoing arguments, the Appellants respectfully solicit the Honorable Board to reverse the Examiner's rejections of claims 1-7, 10, 12-18, 21-25, 27, 31-33, 35, 37-46, 49, 50 and 52-69. In addition, as the Appellants' remarks with respect to the Examiner's rejections are sufficient to overcome the rejections, the Appellants' silence as to assertions by the Examiner in the Final Office or certain requirements that may be applicable to such rejections (e.g., whether a reference constitutes prior art) is not a concession by the Appellants that such assertions are accurate or such requirements have been met, and the Appellants reserve the right to analyze and dispute such in the future.

To the extent necessary, a petition for an extension of time under 37 C.F.R. § 1.136 is hereby made. Please charge any shortage in fees due in connection with the filing of this paper, including extension of time fees, to Deposit Account 50-1070 and please credit any excess fees to such deposit account.

Respectfully submitted,

HARRITY SNYDER, L.L.P.

By: /Glenn Snyder, Reg. No. 41,428/
Glenn Snyder
Reg. No. 41,428

Date: August 28, 2007

11350 Random Hills Road
Suite 600
Fairfax, VA 22030
Telephone: (571) 432-0800
Facsimile: (571) 432-0808

IX. APPENDIX

1. A method for detecting and preventing security breaches in a network, the method comprising:

reassembling a plurality of TCP packets in network traffic into a TCP stream;

grouping the plurality of TCP packets into packet flows and sessions;

storing the packet flows in packet flow descriptors, the packet flow descriptors being addressed by a hash value computed from a 5-tuple comprising a source IP address, a destination IP address, a source port, a destination port and a protocol type;

inspecting the TCP stream to detect information indicative of a security breach;

dropping a TCP packet from the TCP stream if the TCP stream contains information indicative of a security breach;

forwarding a TCP packet from the TCP stream to a network destination if the TCP stream does not contain information indicative of a security breach,

wherein inspecting the TCP stream to detect information indicative of a security breach comprises:

storing a plurality of protocol specifications supported by the network in a protocol database,

querying the protocol database to determine whether the plurality of TCP packets are compliant with one or more of the plurality of protocol specifications in the protocol database, and

searching for a network attack identifier in the TCP stream based on the packet flow descriptors and sessions associated with the TCP stream.

2. The method of claim 1, wherein inspecting the TCP stream to detect information indicative of security breaches comprises inspecting the TCP stream for protocol irregularities.

3. The method of claim 1, wherein inspecting the TCP stream to detect information indicative of a security breach comprises searching the TCP stream for attack signatures.

4. The method of claim 3, wherein searching the TCP stream for attack signatures comprises using stateful signature detection.

5. The method of claim 1, wherein inspecting the TCP stream to detect information indicative of a security breach comprises using a plurality of network intrusion detection methods.

6. The method of claim 5, wherein the plurality of network intrusion detection methods comprises at least protocol anomaly detection.

7. The method of claim 5, wherein the plurality of network intrusion detection methods comprises at least signature detection.

10. The method of claim 1, further comprising searching the packet flow descriptors for traffic signatures.

12. The method of claim 1, wherein the network attack identifier comprises a protocol irregularity.

13. The method of claim 1, wherein the network attack identifier comprises an attack signature.

14. The method of claim 1, wherein the network attack identifier comprises a plurality of network attack identifiers.

15. The method of claim 14, wherein the plurality of network attack identifiers comprises at least a protocol irregularity.

16. The method of claim 14, wherein the plurality of network attack identifiers comprises at least an attack signature.

17. The method of claim 13, wherein the attack signature and traffic signatures are stored in a signatures database.

18. A method comprising:

reassembling a plurality of TCP packets into a TCP stream;

inspecting the TCP stream to detect information indicative of a security breach;

dropping a TCP packet from the TCP stream if the TCP stream contains information

indicative of a security breach;

forwarding a TCP packet from the TCP stream to a network destination if the TCP stream does not contain information indicative of a security breach; and

grouping the plurality of TCP packets into packet flows and sessions, wherein grouping the plurality of TCP packets into packet flows and sessions comprises storing the packet flows and sessions in a hash table,

wherein inspecting the TCP stream to detect information indicative of a security breach comprises:

storing the packet flows in packet flow descriptors, and

searching for a network attack identifier in the TCP stream based on the packet flow descriptors and sessions associated with the TCP stream, wherein storing the packet flows and sessions in a hash table comprises computing a hash value from a 5-tuple comprising a source IP address, a destination IP address, a source port, a destination port and a protocol type.

21. The method of claim 3, wherein searching the TCP stream for attack signatures comprises querying a signatures database to determine whether there are matching signatures in the TCP stream.

22. A method for detecting and preventing security breaches in a network, the method comprising:

reassembling a plurality of TCP packets into a TCP stream;

inspecting the TCP stream to detect information indicative of a security breach;

dropping a TCP packet from the TCP stream if the TCP stream contains information indicative of a security breach;

forwarding a TCP packet from the TCP stream to a network destination if the TCP stream does not contain information indicative of a security breach, wherein inspecting the TCP stream to detect information indicative of a security breach comprises:

querying a signatures database to determine whether there are matching signatures in the TCP stream using deterministic finite automata for pattern matching.

23. The method of claim 1, further comprising reconstructing the plurality of TCP packets from a plurality of packet fragments.

24. A system for detecting and preventing security breaches in a network, the system comprising:

a TCP reassembly software module for reassembling a plurality of TCP packets in network traffic into a TCP stream;

a software module for inspecting the TCP stream to detect information indicative of a security breach;

a software module for dropping a TCP packet from the TCP stream if the TCP stream contains information indicative of a security breach;

a software module for forwarding a TCP packet from the TCP stream to a network destination if the TCP stream does not contain information indicative of a security breach; and

at least one processing device configured to execute the TCP reassembly software

module, the software module for inspecting the TCP stream, the software module for dropping a TCP packet and the software module for forwarding a TCP packet,

wherein the software module for inspecting the TCP stream comprises at least a protocol anomaly detection software module and a flow manager software module, the protocol anomaly detection software module comprising:

a routine for storing a plurality of protocol specifications supported by the network in a protocol database, and

a routine for querying the protocol database to determine whether the plurality of TCP packets are compliant with one or more of the plurality of protocol specifications in the protocol database,

wherein the flow manager software module is configured to:

group the plurality of TCP packets into packet flows and sessions,

store the packet flows in packet flow descriptors, the packet flow descriptors being addressed by a hash value computed from a 5-tuple comprising a source IP address, a destination IP address, a source port, a destination port and a protocol type, and

search for a network attack identifier in the TCP stream based on the packet flow descriptors and sessions associated with the TCP stream.

25. The system of claim 24, further comprising an IP defragmentation software module for reconstructing a plurality of packet fragments into the plurality of TCP packets.

27. A system comprising:

a TCP reassembly software module for reassembling a plurality of TCP packets network traffic into a TCP stream;

a software module for inspecting the TCP stream to detect information indicative of a security breach;

a software module for dropping a TCP packet from the TCP stream if the TCP stream contains information indicative of a security breach;

a software module for forwarding a TCP packet from the TCP stream to a network destination if the TCP stream does not contain information indicative of a security breach;

a flow manager software module for grouping the plurality of TCP packets into packet flows and sessions, wherein the flow manager software module comprises a routine for storing the packet flows and sessions into a hash table, storing the packet flows in packet flow descriptors, and searching for a network attack identifier in the TCP stream based on the packet flow descriptors and sessions associated with the TCP stream; and

at least one processing device configured to execute the TCP reassembly software module, the software module for inspecting the TCP stream, the software module for dropping a TCP packet, a software module for forwarding a TCP packet and the flow manager software module,

wherein the routine for storing the packet flows and sessions into a hash table comprises a routine for computing a hash value from a 5-tuple comprising a source IP address, a destination IP address, a source port, a destination port and a protocol type.

31. The system of claim 24, wherein the software module for inspecting the TCP stream to detect information indicative of a security breach comprises a stateful signature detection software module.

32. The system of claim 27, further comprising a traffic signature detection software module for searching the packet flow descriptors for traffic signatures.

33. The system of claim 24, wherein the software module for inspecting the TCP stream for information indicative of a security breach comprises a plurality of software modules.

35. The system of claim 33, wherein the plurality of software modules comprises at least a stateful signature detection software module.

37. The system of claim 24, wherein the protocol specifications comprise specifications of one or more of: TCP protocol; HTTP protocol; SMTP protocol; FTP protocol; NETBIOS protocol; IMAP protocol; POP3 protocol; TELNET protocol; IRC protocol; RSH protocol; REXEC protocol; and RCMD protocol.

38. The system of claim 35, wherein the stateful signature detection software module comprises a routine for querying a signatures database to determine whether there are matching attack signatures in the TCP stream.

39. A system for detecting and preventing security breaches in a network, the system comprising:

a TCP reassembly software module for reassembling a plurality of TCP packets in network traffic into a TCP stream;

a software module for inspecting the TCP stream to detect information indicative of a security breach;

a software module for dropping a TCP packet from the TCP stream if the TCP stream contains information indicative of a security breach; ~~and~~

a software module for forwarding a TCP packet from the TCP stream to a network destination if the TCP stream does not contain information indicative of a security breach, wherein the software module for inspecting the TCP stream to detect information indicative of a security breach comprises a stateful signature detection software module, the stateful signature detection software module comprising:

a routine for querying a signatures database to determine whether there are matching attack signatures in the TCP stream using deterministic finite automata for pattern matching; and

at least one processing device configured to execute the TCP reassembly software module, the software module for inspecting the TCP stream, the software module for dropping a TCP packet and the software module for forwarding a TCP packet.

40. The system of claim 24, further comprising:

a routine for collecting a plurality of security logs and alarms recording information about security breaches found in the TCP stream;

a routine for storing a network security policy identifying the network traffic to inspect and a plurality of network attacks to be detected and prevented;

a routine for distributing the network security policy to one or more gateway points in the network; and

a routine for updating the protocol database and a signatures database.

41. The system of claim 24, further comprising a graphical user interface comprising:

a routine for displaying network security information to network security administrators; and

a routine for specifying a network security policy.

42. A system for detecting and preventing security breaches in a network, the system comprising:

a network intrusion detection and prevention sensor located in a network gateway, wherein the network intrusion detection and prevention sensor including at least one processor configured to execute:

a routine for reassembling a plurality of TCP packets into a TCP stream;

a software module for inspecting the TCP stream to detect information indicative of a security breach, wherein inspecting the TCP stream to detect information indicative of a security breach comprises:

storing a plurality of protocol specifications supported by the network in a protocol database, and

querying the protocol database to determine whether the plurality of TCP packets are compliant with one or more of the plurality of protocol specifications in the protocol database;

a software module for dropping a TCP packet from the TCP stream if the TCP stream contains information indicative of a security breach; and

a software module for forwarding a TCP packet from the TCP stream to a network destination if the TCP stream does not contain information indicative of a security breach;

a central management server to control the network intrusion detection and prevention sensor; and

a graphical user interface for configuring the network intrusion detection and prevention sensor,

wherein the network intrusion detection and prevention sensor further comprises a flow manager software module and a traffic signature detection module, the flow manager software module being configured to:

group the plurality of TCP packets into packet flows and sessions, and

store the packet flows in packet flow descriptors, the packet flow descriptors being addressed by a hash value computed from a 5-tuple comprising a source IP address, a destination IP address, a source port, a destination port and a protocol type, and

the traffic signature detection module is configured to:

search for a network attack identifier in the TCP stream based on the packet flow descriptors and sessions associated with the TCP stream.

43. The system of claim 42, wherein the network intrusion detection and prevention sensor is located within a firewall.

44. The system of claim 42, wherein the network intrusion detection and prevention sensor is located outside a firewall.

45. The system of claim 42, wherein the network intrusion detection and prevention sensor further comprises an IP defragmentation software module for reconstructing a plurality of packet fragments into the plurality of TCP packets.

46. The system of claim 42, wherein the network intrusion detection and prevention sensor further comprises an IP router software module for routing a TCP packet from the TCP stream if the TCP stream does not contain information indicative of a network security breach through the network.

49. The system of claim 42, wherein the software module for inspecting information indicative of a security breach comprises a protocol anomaly detection software module.

50. The system of claim 42, wherein the software module for inspecting information indicative of a security breach comprises a stateful signature detection software module.

52. The system of claim 42, wherein the software module for inspecting information

indicative of a security breach comprises a plurality of software modules.

53. The system of claim 52, wherein the plurality of software modules comprises at least a protocol anomaly detection software module.

54. The system of claim 52, wherein the plurality of software modules comprises at least a stateful signature detection software module.

55. The system of claim 42, wherein the central management server comprises:

a routine for collecting a plurality of security logs and alarms recording information about security breaches found in the TCP stream;

a routine for storing a network security policy identifying the network traffic to inspect and a plurality of network attacks to be detected and prevented; and

a routine for distributing the network security policy to the network intrusion detection and prevention sensor.

56. The system of claim 42, wherein the graphical user interface comprises:

a routine for displaying network security information to network security administrators;

a routine for displaying status information on the network intrusion detection and prevention sensor; and

a routine for specifying a network security policy.

57. A network intrusion detection and prevention sensor for detecting and preventing network security breaches at a network gateway, the network intrusion detection and prevention sensor comprising:

a flow manager software module for grouping a plurality of packets into packet flows and sessions;

a TCP reassembly software module for reassembling a plurality of TCP packets from the plurality of packets into a TCP stream;

a software module for inspecting the TCP stream according to the packet flows and sessions to detect information indicative of a security breach, wherein inspecting the TCP stream to detect information indicative of a security breach comprises:

storing a plurality of protocol specifications supported by the network in a protocol database, and

querying the protocol database to determine whether the plurality of TCP packets are compliant with one or more of the plurality of protocol specifications in the protocol database;

a software module for dropping a packet from the plurality of packets if the TCP stream contains information indicative of a security breach;

a software module for forwarding a packet from the plurality of packets to a network destination if the TCP stream does not contain information indicative of a security breach;

a software module for grouping the plurality of TCP packets into packet flows and sessions and storing the packet flows in packet flow descriptors, the packet flow descriptors being addressed by a hash value computed from a 5-tuple comprising a source IP address, a

destination IP address, a source port, a destination port and a protocol type;

a software module for searching for a network attack identifier in the TCP stream based on the packet flow descriptors and sessions associated with the TCP stream; and

at least one processing device configured to execute the flow manager software module, the TCP reassembly software module, the software module for inspecting the TCP stream, the software module for dropping a packet, a software module for forwarding a packet, a software module for grouping the plurality of TCP packets and the software module for searching for a network attack identifier.

58. The network intrusion detection and prevention sensor of claim 57, further comprising an IP defragmentation software module for reconstructing a plurality of packet fragments into the plurality of TCP packets.

59. The network intrusion detection and prevention sensor of claim 57, wherein the network intrusion detection and prevention sensor further comprises an IP router software module for routing a TCP packet from the TCP stream if the TCP stream does not contain information indicative of a network security breach through the network.

60. The network intrusion detection and prevention sensor of claim 57, wherein the network intrusion detection and prevention sensor is controlled by a network security policy specifying the network traffic to inspect and a plurality of network attacks to be detected and prevented.

61. The network intrusion detection and prevention sensor of claim 60, wherein the network security policy is defined by a network security administrator using a graphical user interface.

62. The network intrusion detection and prevention sensor of claim 61, wherein the graphical user interface comprises:

a routine for displaying network security information to network security administrators;

a routine for displaying status information on the network intrusion detection and prevention sensor; and

a routine for specifying the network security policy.

63. The network intrusion detection and prevention sensor of claim 60, wherein the security policy is stored and distributed to the network intrusion detection and prevention sensor by a central management server.

64. The network intrusion detection and prevention sensor of claim 63, wherein the central management server comprises a routine for collecting a plurality of security logs and alarms recording information about security breaches found in the TCP stream.

65. The network intrusion detection and prevention sensor of claim 57, wherein the software module for inspecting the TCP stream according to the packet flows and sessions to detect information indicative of a security breach comprises a protocol anomaly detection

software module.

66. The network intrusion detection and prevention sensor of claim 57, wherein the software module for inspecting the TCP stream according to the packet flows and sessions to detect information indicative of a security breach comprises a stateful signature detection software module.

67. The network intrusion detection and prevention sensor of claim 58, wherein the software module for inspecting the plurality of packets according to the packet flows and sessions to detect information indicative of a security breach comprises a plurality of software modules.

68. The network intrusion detection and prevention sensor of claim 67, wherein the plurality of software modules comprises at least a protocol anomaly detection software module.

69. The network intrusion detection and prevention sensor of claim 67, wherein the plurality of software modules comprises at least a stateful signature detection software module.

X. EVIDENCE APPENDIX

None

XI. RELATED PROCEEDINGS APPENDIX

None